7장. 연산자

7장. 연산자

- **연산자(operator)** : 하나 이상의 표현식을 대상으로 산술, 할당, 비교, 논리, 타입, 지수 연산 등을

  수행해 하나의 값을 만든다. 이때 연산의 대상을 **피연산자(operand)**라 함.

```
// 타입 연산자
typeof 'Hi' // -> string
```

7.1 산술 연산자

1. 산술 연산자(arithmetic operator)는 피연산자를 대상으로 수학적 계산을 수행해 새로운 숫자 값을 만듦.
   - 산술 연산이 불가능한 경우, **NaN(Not a Number)** 반환

7.1.1 이항 산술 연산자

1. 이항 산술 연산자는 피연산자의 값을 변경하는 부수 효과(side effect)가 없다.

   = 어떤 산술 연산을 해도 피연산자의 값이 바뀌는 경우는 언제나 새로운 값을 만듦

7.1.2 단항 산술 연산자

1. **++(증가), —(감소) 연산자는 피연산자의 값을 변경하는 부수 효과가 있다.**
   - ++(증가), —(감소) 연산자는 위치에 의미가 있다.
2. 숫자 타입이 아닌 피연산자에 + 단항 연산자 사용

   → 피연산자를 **숫자 타입으로 변환**하여 반환

   ex) + 단항 연산자

```
x = 'Hello';     // 문자열을 숫자로 타입 변환 불가능 -> NaN 반
console.log(+x); // NaN
// 부수 효과는 X
console.log(x);  // "Hello"
```

7.1.3 문자열 연결 연산자

1. +연산자는 피연산자 중 하나 이상이 문자열인 경우, 문자열 연결 연산자로 동작

```
'1' + 2; // '12'
1 + '2'; // '12'

1 + true; // 2  (true는 1로 타입 변환됨.)
1 + null; // 1  (null, false는 0으로 타입 변환됨.)

+undefined;    // NaN
1 + undefined; // NaN
```

❗ **주목할 부분**

- 개발자의 의도와 상관없이 JS 엔진에 의해 암묵적으로 타입이 자동 변환되기도 함.

  이를 **암묵적 타입 변환(implicit coercion), 타입 강제 변환(type coercion)** 이라고 함.

7.2 할당 연산자

1. **할당 연산자(assignment operator)**는 좌항의 변수에 값을 할당하므로,

   변수 값이 변하는 부수효과가 있다.

7.3 비교 연산자

7.3.1 동등/일치 비교 연산

1. **동등 비교(==) 연산자는 좌항 & 우항의 피연산자 비교할 때, 먼저 암묵적 타입 변환을 통해**

   타입을 일치시킨 후, 같은 값인지 비교한다.

```
// 동등 비교
5 == 5; // -> true

// 타입은 다르지만 암묵적 타입 변환을 통해 타입을 일치시키면 동등하다.
5 == '5'; // -> true
```

2. **일치 비교(===) 연산자는 좌항 & 우항 피연산자가 타입도 같고, 값도 같은 경우에 한하여**

   true를 반환한다.

```
// 일치 비교
5 === 5; // -> true

5 === '5'; // -> false
```

```
// NaN은 자신과 일치하지 않는 유일한 값이다.
NaN === NaN; // false
```

3. 빌트인 함수로 NaN을 조사할 수 있다.

```
Number.isNaN(NaN);            // true
Number.isNaN(10);             // false
Number.isNaN(1 + undefined);  // true
```

4. 숫자 0도 주의❗

```
0 === -0;  // true
0 == -0;   // true
```

5. **Object.is** 메서드

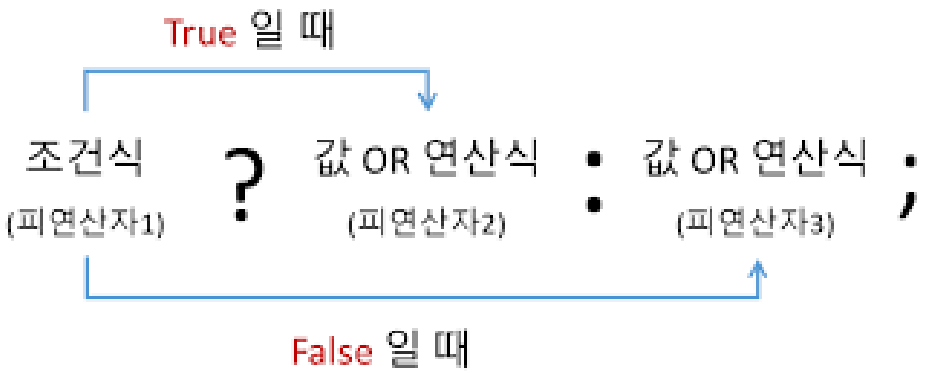
```
-0 === +0;           // true
Object.is(-0, +0);   // false

NaN === NaN;         // false
Object.is(NaN, NaN)  // true
```

7.3.2 대소 관계 비교 연산자

1. >, <, ≥, ≤ 부수효과 X

7.4 삼항 조건 연산자



1. 조건식 : Boolean 타입의 값으로 평가될 표현식

2. if ... else 문과의 차이점

- **삼항 조건 연산자 표현식은 값으로 평가할 수 있는 표현식인 문**이다.

7.5 논리 연산자

1. ||(OR), &&(AND), !(NOT) - 부수효과 X

7.6 쉼표 연산자

```
var x, y, z;
x = 1, y = 2, z = 3; // 마지막 피연산자의 3 반환
```

7.7 그룹 연산자

1. 소괄호( ) → 자신의 피연산자인 표현식을 가장 먼저 평가

- 연산자 우선순위가 가장 높다.

7.8 typeof 연산자

1. `typeof null  // “object" 반환` - 자바스크립트의 첫 번째 버전의 버그
    - 기존 코드에 영향을 줄 수 있어 아직까지 수정되지 못하고 있음.

7.9 지수 연산자

1. 지수 연산자의 결합 순서는, 우항에서 좌항. 즉, 우결합성을 갖는다.

2. 음수의 경우 **괄호**로 묶어야 함.

```
-5 ** 2     // SyntaxError
(-5) ** 2;  // 25
```

7.10 그 외의 연산자

7.11 연산자의 부수 효과

1. 부수 효과가 있는 연산자 : 할당 연산자(=), 증가/감소 연산자(++/—), delete 연산자

```
var o = { a: 1 };

delete o.a;
console.log(o); // {}
```

7.12 연산자 우선순위

우선순위	연산자
1	`( )`
2	`new`(매개변수 존재), `.`, `[ ]`(프로퍼티 접근), `( )`(함수 호출), `?.`(옵셔널 체이닝 연산자[9])
3	`new`(매개변수 미존재)
4	`x++`, `x--`
5	`!x`, `+x`, `-x`, `++x`, `--x`, `typeof`, `delete`
6	`**`(이항 연산자 중에서 우선순위가 가장 높다)
7	`*`, `/`, `%`
8	`+`, `-`
9	`<`, `<=`, `>`, `>=`, `in`, `instanceof`
10	`==`, `!=`, `===`, `!==`

우선순위	연산자
11	`??`(null 병합 연산자[10])
12	`&&`
13	`||`
14	`? ... : ...`
15	할당 연산자(`=`, `+=`, `-=`, ...)
16	`,`

7.13 연산자 결합 순서

1. 연산자의 어느 쪽(좌항 or 우항)부터 평가를 수행할 것인지 나타내는 순서를 의미.